CLAIM AMENDMENTS

1. (currently amended) A method for verifying the trustworthiness of executable Web browser software, said method comprising~~, in the following order,~~ the steps of:

transmitting via a distributed network an electronic document requiring a digital signature from a first user computer to a second user computer;

electronically signing the electronic document by executable browser software at the second user computer to create a first digital signature, wherein the first digital signature ~~includes as an attribute~~ is affixed to a combination comprising content plus a second digital signature ~~serving to verify the trustworthiness of the executable browser software itself~~, wherein the second digital signature ~~verifying~~ is affixed to, and thus verifies the authenticity of, at least one executable component running in an environment of the executable browser software;

transmitting the signed electronic document from the second user computer to the first user computer; and

subsequent to the two transmitting steps and the electronically signing step, authenticating the second digital signature.

2. (previously presented) The method of claim 1, further comprising the step of determining whether the entity that executed the second digital signature is authorized to certify the trustworthiness of the at least one component.

3. (canceled)

4. (canceled)

5. (previously presented) The method of claim 1, wherein the authenticating step comprises verifying the authenticity of the second digital signature.

6. (previously presented) The method of claim 5, wherein the authenticity of the second digital signature is verified by checking a digital certificate associated with said second digital signature.

7. (currently amended) The method of claim 1, wherein ~~the first digital signature~~ said combination further comprises ~~as an attribute~~ a hash of the at least one component running in the

executable browser software environment, said method further comprising the step of comparing said hash to a known-good hash of the at least one component .

8. (previously presented) The method of claim 1, wherein the authenticating step is performed by the first user computer.

9. (previously presented) The method of claim 1, wherein the authenticating step is performed by a computer maintained by a financial institution participant.

10. (previously presented) The method of claim 1, wherein the authenticating step is performed by an independent entity that is not a financial institution participant.

11. (previously presented) The method of claim 1, wherein the authenticating step is performed by the second user computer.

12. (currently amended) The method of claim 1, wherein said combination further comprises an unsigned executable component running in the executable browser software environment of the second user computer ~~is also included as an attribute of the first digital signature~~.

13. (previously presented) The method of claim 12, wherein the unsigned executable component is copied from RAM of the second user computer.

14. (previously presented) The method of claim 12, wherein the unsigned executable component is copied from non-volatile memory of the second user computer.

15. (canceled)

16. (canceled)

17. (canceled)

18. (previously presented) A method for verifying the trustworthiness of an executable Web browser software module, said method comprising the steps of:

creating a first set of known-good hashes, the first set of known-good hashes comprising: a hash of the Web browser software module at a first point in time, and

a plurality of hashes corresponding to a plurality of executable browser components at the first point in time;

determining the status of the Web browser software module at a second point in time by creating a second set of hashes, the second set of hashes comprising:

a hash of the Web browser software module at the second point in time, and

a plurality of hashes corresponding to the plurality of executable browser components at the second point in time;

verifying the second set of hashes to ensure that each hash was created by a trusted source; and

subsequent to the executable Web browser software module having executed by virtue of having digitally signed an electronic document, comparing the first set of hashes to the second set of hashes to determine the trustworthiness of the Web browser software module.

19. (previously presented) The method of claim 18, wherein the second point in time is subsequent to the first point in time.

20. (previously presented) The method of claim 18, wherein the determining step further comprises verifying the status of the Web browser software module when the first set of hashes matches the second set of hashes.

21. (previously presented) The method of claim 18, wherein the determining step further comprises determining that the status of the Web browser software module is bad when the first set of hashes does not match the second set of hashes.

22. (previously presented) The method of claim 18, wherein the determining step further comprises determining that the status of the Web browser software module is unknown when it is not determined that a hash in the second set of hashes was created by a trusted source.

23. (previously presented) The method of claim 18, wherein the determining step further comprises determining that the status of the Web browser software module is unknown when it is determined that a hash in the second set of hashes was not created by a trusted source.

24. (previously presented) The method of claim 18, wherein the first set of hashes is maintained by a trusted entity, said method further comprising the steps of:

receiving from a requestor a request to determine the trustworthiness of the Web browser software module, the request including the second set of hashes;

generating a report about the status of the Web browser software module based on a result of the determining step; and

transmitting the report to the requestor.

25. (original) The method of claim 24, wherein the steps of receiving, determining, generating, and transmitting are performed by the trusted entity.

26. (previously presented) The method of claim 18, wherein the second set of hashes comprises at least one hash of executable browser components at a second point in time.

27. (original) The method of claim 26, wherein the first set of hashes comprises hashes at a first point in time corresponding to the hashes in the second set of hashes.

28. (previously presented) The method of claim 27, wherein the second point in time is subsequent to the first point in time.

29. (previously presented) The method of claim 27, wherein at least one of the hashes in the second set of hashes has been signed by a trusted source.

30. (previously presented) The method of claim 29, wherein the verifying step further comprises verifying that a hash in the second set of hashes was created by a trusted source, by verifying a signature on the hash.

31. (previously presented) The method of claim 18, wherein the browser status request is received from a first customer seeking to verify the trustworthiness of the Web browser software module running on a computer in the possession of a second customer.

32. (previously presented) The method of claim 31, wherein the first customer and the second customer are parties to a transaction.

33. (original) The method of claim 32, wherein the first customer is a buyer and the second customer is a seller in the transaction.

34. (previously presented) The method of claim 31, wherein the second customer disaffirms the transaction based on the status of the Web browser software module.

35. (previously presented) Apparatus for providing trusted Web browser verification, said apparatus comprising:

a trusted verifier module;

coupled to the trusted verifier module, means for maintaining by the trusted verifier module a first set of hashes generated by a microprocessor, the first set of hashes comprising a first hash of an executable software browser and a first plurality of hashes corresponding to a plurality of executable browser components, the first set of hashes being a known-good set of hashes;

coupled to the trusted verifier module, means for receiving by the trusted verifier module a browser status request, the browser status request including a second set of hashes generated by a microprocessor, the second set of hashes comprising a second hash of the browser and a second plurality of hashes corresponding to a plurality of executable browser components;

coupled to the trusted verifier module, means for verifying by the trusted verifier module that each hash in the second set of hashes was created by a trusted source; and

coupled to the trusted verifier module, means for determining by the trusted verifier module the status of the browser, subsequent to the browser having executed by virtue of having digitally signed an electronic document, based on the first set of hashes and the second set of hashes.

36. (previously presented) The apparatus of claim 35, wherein the trusted verifier module determines the status of the browser by comparing the first set of hashes with the second set of hashes.

37. (previously presented) The apparatus of claim 36, wherein the trusted verifier module verifies the status of the browser when the first set of hashes matches the second set of hashes.

38. (previously presented) The apparatus of claim 36, wherein the means for determining determines that the status of the browser is bad when the first set of hashes does not match the second set of hashes.

39. (previously presented) The apparatus of claim 36, wherein the means for determining determines that the status of the browser is unknown when it is not determined that a hash in the second set of hashes was created by a trusted source.

40. (previously presented) The apparatus of claim 36, wherein the means for determining determines that the status of the browser is unknown when it is determined that a hash in the second set of hashes was not created by a trusted source.

41. (previously presented) The apparatus of claim 35, wherein the second set of hashes comprises at least one hash of executable browser components at a second point in time.

42. (previously presented) The apparatus of claim 41, wherein the first set of hashes comprises hashes at a first point in time corresponding to the hashes in the second set of hashes.

43. (previously presented) The apparatus of claim 42, wherein the determining means is invoked at a second point in time subsequent to the first point in time.

44. (previously presented) The apparatus of claim 42, wherein at least one of the hashes in the second set of hashes has been signed by a trusted source.

45. (previously presented) The apparatus of claim 44, wherein the means for verifying verifies that a hash in the second set of hashes was created by a trusted source, by verifying the signature on the hash.

46. (previously presented) The apparatus of claim 35, wherein the browser status request is received from a first customer seeking to verify the trustworthiness of a browser running on a computer in the possession of a second customer.

47. (previously presented) The apparatus of claim 46, wherein the first customer and the second customer are parties to a transaction.

48. (previously presented) The apparatus of claim 47, wherein the first customer is a buyer and the second customer is a seller in the transaction.

49. (previously presented) The apparatus of claim 46, wherein the second customer disaffirms the transaction based on the status of the browser.

50. (previously presented) In conjunction with apparatus comprising a root entity, a first participant, a second participant, a first customer of the first participant, and a second customer of the second participant, a method for verifying the trustworthiness of an executable Web browser in possession of the first customer, said method comprising:

a) maintaining at a trusted verifier module a first set of hashes, the first set of hashes comprising a first hash of the browser;

b) generating by the first customer a second set of hashes, the second set of hashes comprising a second hash of the browser;

c) transmitting by the first customer the second set of hashes to the second customer, using a network connection;

d) generating by the second customer a browser status request, the browser status request including the second set of hashes;

e) transmitting by the second customer the browser status request to the second participant;

f) forwarding by the second participant the browser status request to the trusted verifier module;

g) determining by the trusted verifier module a status of the browser;

h) generating by the trusted verifier module a browser status response;

i) forwarding by the trusted verifier module the browser status response to the second participant; and

j) transmitting by the second participant the browser status response to the second customer.

51. (previously presented) The method of claim 50, wherein the trusted verifier module determines the status of the browser by comparing the first set of hashes with the second set of hashes.

52. (previously presented) The method of claim 51, wherein the status of the browser is verified when the first set of hashes matches the second set of hashes.

53. (original) The method of claim 51, wherein the status of the browser is one of good, bad, or unknown.

54. (previously presented) The method of claim 50, wherein the trusted verifier module verifies that each hash in the second set of hashes was created by a trusted source.

55. (previously presented) The method of claim 54, wherein the status of the browser is verified when the first set of hashes matches the second set of hashes.

56. (original) The method of claim 54, wherein the status of the browser is one of good, bad, or unknown.

57. (previously presented) The method of claim 50, wherein the first customer and the second customer are parties to a transaction.

58. (original) The method of claim 57, wherein the first customer is a buyer and the second customer is a seller in the transaction.

59. (original) The method of claim 58, wherein the second customer disaffirms the transaction based on the status of the browser.

60. (original) The method of claim 50, wherein the root entity establishes a set of operating rules for the system.

61. (original) The method of claim 50, wherein the first participant is a financial institution.

62. (original) The method of claim 50, wherein the second participant is a financial institution.

63. (original) The method of claim 50, wherein the first participant comprises a transaction coordinator for processing browser status requests.

64. (original) The method of claim 50, wherein the second participant comprises a transaction coordinator for processing browser status requests.

65. (previously presented) The method of claim 50, wherein the trusted verifier module is an integrated component of the first participant.

66. (previously presented) The method of claim 50, wherein the trusted verifier module is an integrated component of the second participant.

67. (previously presented) The method of claim 50, wherein the trusted verifier module is a distinct entity from the first and second participants.

68. (previously presented) Apparatus for verifying the trustworthiness of an executable software browser in possession of a first customer, said apparatus comprising:

a root entity;

a first participant;

a second participant;

a first customer of the first participant;

a second customer of the second participant;

means for maintaining at a trusted verifier module a first set of hashes, the first set of hashes comprising a first hash of the browser;

means for generating by the first customer a second set of hashes, the second set of hashes comprising a second hash of the browser;

means for transmitting by the first customer the second set of hashes to the second customer using a network connection;

means for generating by the second customer a browser status request, the browser status request including the second set of hashes;

means for transmitting by the second customer the browser status request to the second participant;

means for forwarding by the second participant the browser status request to the trusted verifier module;

means for determining by the trusted verifier module a status of the first customer's browser;

means for generating by the trusted verifier module a browser status response;

means for forwarding by the trusted verifier module the browser status response to the second participant; and

means for transmitting by the second participant the browser status response to the second customer.

69. (previously presented) The apparatus of claim 68, wherein the trusted verifier module determines the status of the browser by comparing the first set of hashes with the second set of hashes.

70. (previously presented) The apparatus of claim 69, wherein the status of the browser is verified when the first set of hashes matches the second set of hashes.

71. (previously presented) The apparatus of claim 69, wherein the status of the browser is one of good, bad, or unknown.

72. (previously presented) The apparatus of claim 68, wherein the trusted verifier module verifies that each hash in the second set of hashes was created by a trusted source.

73. (previously presented) The apparatus of claim 72, wherein the trusted verifier module determines the status of the browser by comparing the first set of hashes with the second set of hashes.

74. (previously presented) The apparatus of claim 73, wherein the status of the browser is verified when the first set of hashes matches the second set of hashes.

75. (previously presented) The apparatus of claim 74, wherein the status of the browser is one of good, bad, or unknown.

76. (previously presented) The apparatus of claim 68, wherein the first customer and the second customer are parties to a transaction.

77. (previously presented) The apparatus of claim 68, wherein the first customer is a buyer and the second customer is a seller in the transaction.

78. (previously presented) The apparatus of claim 68, wherein the second customer disaffirms the transaction based on the status of the browser.

79. (previously presented) The apparatus of claim 68, wherein the root entity establishes a set of operating rules for the system.

80. (previously presented) The apparatus of claim 68, wherein the first participant is a financial institution.

81. (previously presented) The apparatus of claim 68, wherein the second participant is a financial institution.

82. (previously presented) The apparatus of claim 68, wherein the first participant comprises a transaction coordinator for processing browser status requests.

83. (previously presented) The apparatus of claim 68, wherein the second participant comprises a transaction coordinator for processing browser status requests.

84. (previously presented) The apparatus of claim 68, wherein the trusted verifier module is an integrated component of the first participant.

85. (previously presented) The apparatus of claim 68, wherein the trusted verifier module is an integrated component of the second participant.

86. (previously presented) The apparatus of claim 68, wherein the trusted verifier module is a distinct entity from the first and second participants.